IN THE CLAIMS

Please amend the claims as follows:

Claims 1-8. (Cancelled).

9. (Currently Amended) A method comprising:

performing a run time dependency check during a current iteration using prior computed values obtained from a predetermined number of previous adjacent iterations in a sparse array matrix; [[and]]

parallelizing dependent loop instructions between the current iteration and subsequent multiple iterations by using the computed values to make the dependence loop instructions into independent loop iterations so as to increase instruction-level parallelism and reduce recurrence initiation interval in the current iteration as a function of the run time dependency check during the current iteration; and

computing a virtual unrolling factor using the recurrence initiation interval and latency in number of cycles of a floating point multiply add operation in the sparse array matrix, wherein the prior computed values are based on virtual unrolling using the computed virtual unrolling factor.

10. (Cancelled).

11. (Currently Amended) The method of claim [[10]] 9, wherein the virtual unrolling factor includes three previous iterations.

12. (Cancelled).

13. (Original) The method of claim 9, further comprising:

assigning the prior computed values to a predetermined number of adjacent registers.

14. (Original) The method of claim 12, further comprising:

performing register rotation to include computed values obtained from the current iteration; and

repeating performing the run time dependency check and parallelizing the dependence loop instructions to increase the instruction-level parallelism and reduce the recurrence initiation interval in a next iteration.

15. (Currently Amended) A method comprising:

transforming sparse array matrix code to perform a run time dependency check using a predetermined number of prior computed values;

software-pipelining the transformed sparse array matrix code to perform the run time dependency check in a current iteration using the predetermined number of prior computed values; [[and]]

software-pipelining to parallelize multiple iterations by overlapping execution of dependence loop instructions in the prior computed values to reduce recurrence initiation interval in the sparse array matrix based on the run time dependency check; and

computing a virtual unrolling factor using the recurrence initiation interval and latency in number of cycles of a floating point multiply add operation in the sparse array matrix, wherein the predetermined number of prior computed values is based on virtual unrolling using the computed virtual unrolling factor.

16. (Currently Amended) The method of claim 15, wherein software-pipelining the transformed sparse array matrix code comprises:

computing a predetermined number of variables based on [[a]] the virtual unrolling factor;

initializing the computed predetermined number of variables;

loading the prior computed values into the predetermined number of variables;

assigning the prior computed values to a predetermined number of substantially adjacent registers; and

software-pipelining using the assigned prior computed values.

17. (Previously Presented) The method of claim 16, further comprising:

performing register rotation to include computed values obtained from the current iteration; and

repeating the software-pipelining and using the register rotated computed values to reduce the recurrence initiation interval in a next iteration.

18. (Currently Amended) The method of claim 17, wherein the computed values obtained from the predetermined number of prior adjacent iterations are based on virtual unrolling by using [[a]] the virtual unrolling factor.

19. (Original) The method of claim 18, wherein the virtual unrolling factor is three.

20. (Currently Amended) A method comprising:

transforming loop computations from a predetermined number of prior adjacent iterations in sparse arrays/matrices to current loads code to perform a run time dependency check using a predetermined number of prior computed values;

software-pipelining the transformed loop computations from the predetermined number of prior adjacent iterations to perform a run time dependency check in a current iteration using the predetermined number of prior computed values; [[and]]

parallelizing the loop computations using the prior computed values; and

computing a virtual unrolling factor using recurrence initiation interval and latency in number of cycles of a floating point multiply add operation in the sparse arrays/matrices, wherein the predetermined number of prior computed values is based on virtual unrolling using the computed virtual unrolling factor.

21. (Currently Amended) The method of claim 20, wherein the predetermined number of prior computed values is obtained from the predetermined number of prior adjacent iterations based on [[a]] the virtual unrolling factor.

22. (Original) The method of claim 21, wherein the virtual unrolling factor is three.

23. (Original) The method of claim 20, wherein parallelizing the computations using the prior computed values comprise:

overlapping execution of dependence loop instructions in multiple dependent iterations in the sparse arrays/matrices using the software-pipelining.

24. (Original) The method of claim 20, further comprising:

assigning the prior computed values to a predetermined number of adjacent register.

25. (Original) The method of claim 24, further comprising:

performing register rotation to include computed values obtained from the current iteration.

26. (Currently Amended) An article comprising a computer-readable medium which stores computer-executable instructions, the instructions causing a computer to:

~~performing~~ perform a run time dependency check during a current iteration using prior computed values obtained from a predetermined number of previous adjacent iterations in a sparse array matrix; [[and]]

~~parallelizing~~ parallelize dependent loop instructions between the current and subsequent multiple iterations by using the computed values to make the dependence loop instructions into independent loop iterations; and

compute a virtual unrolling factor using recurrence initiation interval and latency in number of cycles of a floating point multiply add operation in the sparse array matrix, wherein the prior computed values are based on virtual unrolling using the computed virtual unrolling factor.

27. (Currently Amended) The article comprising a computer-readable medium which stores the computer executable instruction of claim 26, wherein instructions causing a

computer to perform ~~performing~~ the run time dependency check <u>further comprise</u> instructions to ~~comprises~~:

~~transforming~~ <u>transform</u> loop computations from a predetermined number of prior adjacent iterations in sparse arrays/matrices to current loads code to perform a run time dependency check using a predetermined number of prior computed values; and

~~software-pipelining~~ <u>software-pipeline</u> the transformed loop computations from the predetermined number of prior adjacent iterations to perform a run time dependency check in a current iteration using the predetermined number of prior computed values.

28. (Previously Presented) The article comprising a computer-readable medium which stores the computer-executable instructions of claim 26, wherein the instructions further cause the computer to assign the prior computed values to a predetermined number of adjacent register.

29. (Previously Presented) The article comprising a computer-readable medium which stores the computer-executable instructions of claim 28, wherein the instructions further cause the computer to perform register rotation to include computed values obtained from the current iteration.

30. (Currently Amended) A system comprising:

a bus;

a processor coupled to the bus;

a memory coupled to the processor;

a network interface device;

wherein execution of loop computations in indirectly accessed sparse arrays/matrices by the processor uses software-pipelining to increase instruction-level parallelism and decrease initiation interval by performing:

transforming loop computations from a predetermined number of prior adjacent iterations in sparse arrays/matrices to current loads code to perform a run time dependency check using a predetermined number of prior computed values;

software-pipelining the transformed loop computations from the predetermined number of prior adjacent iterations to perform a run time dependency check in a current iteration using the predetermined number of prior computed values; [[and]]

parallelizing the loop computations using the prior computed values to reduce recurrence initiation interval in the undisambiguated pointer stores from the predetermined number of prior adjacent iterations to current loads code based on the run time dependency check; and

computing a virtual unrolling factor using the recurrence initiation interval and latency in number of cycles of a floating point multiply add operation in the sparse arrays/matrices, wherein the predetermined number of prior computed values is based on virtual unrolling using the computed virtual unrolling factor.

31. (Original) The system of claim 30, wherein the processor further assigns the prior computed values to a predetermined number of adjacent register.

32. (Original) The system of claim 31, wherein the processor further performs register rotation to include computed values obtained from the current iteration.

Claims 33-38. (Cancelled).

39. (New) An article comprising a computer-readable medium which stores computer-executable instructions, the instructions causing a computer to:

perform a run time dependency check during a current iteration using prior computed values obtained from a predetermined number of previous adjacent iterations in a sparse array matrix, the performance of the run time dependency check including instructions to cause the computer to:

transform loop computations from a predetermined number of prior adjacent iterations in sparse arrays/matrices to current loads code to perform a run time dependency check using a predetermined number of prior computed values; and

software-pipeline the transformed loop computations from the predetermined number of prior adjacent iterations to perform a run time dependency check in a current iteration using the predetermined number of prior computed values; and

parallelize dependent loop instructions between the current and subsequent multiple iterations by using the computed values to make the dependence loop instructions into independent loop iterations.

40. (New) The article comprising a computer-readable medium which stores the computer-executable instructions of claim 39, wherein the instructions further cause the computer to assign the prior computed values to a predetermined number of adjacent register.

41. (New) The article comprising a computer-readable medium which stores the computer-executable instructions of claim 40, wherein the instructions further cause the computer to perform register rotation to include computed values obtained from the current iteration.